Amendments to the Claims

Claim 1 (currently amended): A method of serializing software objects, comprising steps of:

creating, for an object to be serialized to a persistent store, a graph structure comprising nodes that embody serializable attributes and values thereof; [[and]]

writing the graph structure to the persistent store as a markup language document, wherein the markup language document reflects one or more original class definitions to which the object for which the graph structure was created adheres; and

deserializing a new instance of the object from the markup language document, further comprising the steps of:

creating a second graph structure from the markup language document;

programmatically determining whether serializable attribute definitions for one or more current class definitions to which the new instance adheres are identical to the serializable attribute definitions for the original class definitions, as reflected in the second graph structure, and if not, performing a programmatic migration of the attribute values in the second graph structure; and

deserializing the new instance from the serializable attributes and values embodied in the second graph structure.

Claim 2 (original): The method according to Claim 1, wherein the markup language document is encoded in Extensible Markup Language ("XML") notation.

Claim 3 (original): The method according to Claim 1, wherein the nodes are written as markup

language elements in the markup language document.

Claim 4 (currently amended): The method according to Claim 1, wherein the nodes reflect a structure of the object according to the one or more original class definitions ~~to which the object adheres~~.

Claim 5 (original): The method according to Claim 4, wherein the structure of the object is reflected in hierarchical relationships among markup language elements of the markup language document.

Claim 6 (original): The method according to Claim 5, wherein the attribute values are reflected in attributes of the markup language elements.

Claims 7 - 8 (canceled)

Claim 9 (currently amended): A method of enabling serialized objects to be preserved following changes to one or more class definitions used in those objects, comprising steps of:

creating, for an object to be serialized to a persistent store, a graph structure comprising nodes that embody a structure of the object and values of serializable attributes of the object;

writing the graph structure to the persistent store, such that serializable information from one or more original class definitions to which the object adheres is persistently captured therein;

programmatically determining, in order to deserialize the persistently captured information

to a new instance of the object, whether serializable attribute definitions for the original class definitions, as reflected in the graph structure, are identical to serializable attribute definitions of one or more current class definitions to which the new instance must adhere; and

deserializing the new instance of the object directly from the serializable information persistently captured within the graph structure, if the programmatically determining step has a positive result, and performing a programmatic migration of the attribute values from the serializable information persistently captured [[with]] within the graph structure otherwise, wherein the programmatic migration further comprises directly accessing individual attribute values from the persistently-captured serializable information.

Claim 10 (original): The method according to Claim 9, wherein the writing step further comprises writing the graph structure to the persistent store as a markup language document.

Claim 11 (canceled)

Claim 12 (currently amended): The method according to Claim [[11]] 9, wherein the directly accessing [[step]] does not require access to a programming language specification of the one or more original class definitions.

Claim 13 (currently amended): A method of deserializing software objects, comprising steps of:

creating, from a markup language document written to a persistent store, a corresponding graph structure, wherein elements of the markup language document and nodes of the

corresponding graph structure embody serializable attributes and values of an object and wherein the markup language document reflects one or more original class definitions to which the object adhered when the markup language document was created; and

deserializing a new instance of the object from the graph structure, further comprising the steps of:

programmatically determining whether serializable attribute definitions for one or more current class definitions to which the new instance adheres are identical to the serializable attribute definitions for the original class definitions, as reflected in the graph structure, and if not, performing a programmatic migration of the attribute values in the graph structure, wherein individual ones of the attribute values are directly accessed from the graph structure; and

deserializing the new instance from the serializable attributes and values embodied in the graph structure.

Claim 14 (canceled)

Claim 15 (currently amended): A data structure for enabling serialized objects to be preserved following changes to one or more class definitions used in those objects, the data structure embodied on a computer-readable medium and comprising a markup language specification of a structure of an object according to one or more original class definitions to which the object adheres and values of serializable attributes of the object, according to the one or more original class definitions, such that the data structure is usable for deserializing a new instance of the object according to one or more current class definitions to which the new instance must adhere

by creating a graph structure from the markup language specification; programmatically determining whether serializable attribute definitions for the one or more current class definitions are identical to serializable attribute definitions for the one or more original class definitions, as reflected in the graph structure, and if not, performing a programmatic migration of the attribute values in the graph structure by directly accessing individual attribute values from the graph structure; and deserializing the new instance from the serializable attributes and values embodied in the graph structure.

Claim 16 (canceled)

Claim 17 (currently amended): A system for serializing software objects, comprising:

means for creating, for an object to be serialized to a persistent store, a graph structure comprising nodes that embody serializable attributes and values thereof; [[and]]

means for writing the graph structure to the persistent store as a markup language document, wherein the markup language document reflects one or more original class definitions to which the object for which the graph structure was created adheres; and

means for deserializing a new instance of the object from the markup language document, further comprising:

means for creating a second graph structure from the markup language document;

means for programmatically determining whether serializable attribute definitions for one or more current class definitions to which the new instance adheres are identical to the serializable attribute definitions for the original class definitions, as reflected in the second graph

structure, and if not, performing a programmatic migration of the attribute values in the second graph structure; and

means for deserializing the new instance from the serializable attributes and values embodied in the second graph structure.

Claim 18 (currently amended): A computer program product for deserializing software objects, the computer program product embodied on one or more computer-readable media and comprising:

computer-readable program code [[means]] for creating, from a markup language document written to a persistent store, a corresponding graph structure, wherein elements of the markup language document and nodes of the corresponding graph structure embody serializable attributes and values of an object and wherein the markup language document reflects one or more original class definitions to which the object adhered when the markup language document was created; and

computer-readable program code [[means]] for deserializing a new instance of the object from the graph structure, further comprising:

computer-readable program code for programmatically determining whether serializable attribute definitions for one or more current class definitions to which the new instance adheres are identical to the serializable attribute definitions for the original class definitions, as reflected in the graph structure, and if not, performing a programmatic migration of the attribute values in the graph structure, wherein individual ones of the attribute values are directly accessed from the graph structure; and

computer-readable program code for deserializing the new instance from the serializable attributes and values embodied in the graph structure.